# Homework 3 - Sentimate Analysis

Kyle Walter

9/7/2021

## Background

For the last part of the three-part series of review for Anna Karenina and The Great Gatsby, I am going to do a sentimate analysis to see how the book's overall sentimate is expressed.

Most of the code for this analysis was given as a framework to complete the Sentimate Analysis with some edits from me to allow the change data structure from Twitter to large Text Corpora.

As discussed in our pervious homework, both coming of age stories are about 75 years apart. One set in a rapidly modernizing Russia and the other 1920s New York.

## Loading the data and building the model

To start off, I called the text for the two Corpora directly from Gutenberg using the package in Python. The command comes with with an aquire and strip the function the basically calls in the full text from Gutenberg, along with some metadata as well. The strip removes the metadata seen at the beginning of the file.

At this point I Tokenized the sentences, breaking up the entire Corpora into indivdual sentences and then within the sentences broke up the text further into individual words. Creating several lists inside of the list with the Corpora data.

The next step was to load in a training dataset. With the plans to train a model to give us sentiment of the stories, I chose to follow the example and use the airline reviews dataset from twitter. The File has 14,640 tweets of passenger reviews. These reviews have been categorized by human readers as either having Postive Sentiment, Negative Sentiment, or Neutral Sentiment.

Once the dataset was successfully loaded into memory, the next step was to remove "@" and "#" symbols that show up with users either mention someone or in this case likely the airline or give keywords that twitter uses to point people similar tweets or topics.

To do this, a string.replace function was used. It reads through the tweets for the aforementioned systems and replaces them with a space. Since two different systems were searched, to minimize the amount of code needed to type they grouped together in a function and then a lambda call done to the text field so that every tweet would be scrubbed.

The next step was to start exploring the training set. First check was to see if the data set was balanced, meaning the sentiment of positive, negative, and neutral are relatively equal. To do this check, the code created three separate data frames by sub-setting for the records in the training set by their respective sentiments.

| | airline_sentiment | counts |
|---|---|---|
| 0 | negative | 9178 |
| 1 | neutral | 3099 |
| 2 | positive | 2363 |

*Initial Sentimate Breakout*

As we can see initially the are almost 3x the amount of negative sentiment, verse either positive or neutral. This presents some problems as a model built on this kind of break out will have a higher likelihood of tagging a sentence with a negative even if the sentence is objectively positive.

One way to fix this is to balance the data set. I choose to use a statistic method known as sampling with replace, which means I will create a new data set from the records of the starting set. I will select records at random and replace the record into the data set so it can be called again.

Pandas offers a method on the data frame called sample for this very task. It also has an argument that will allow for weights to be applied to make some records more likely to be picked than others. I chose to use this in order to set the weights higher for picking neutral and positive over negative. The question was how much higher? I took a normalized value count of the airline sentiment to see which percent of the records negative, positive, and neutral.

```
negative    0.626913
neutral     0.211680
positive    0.161407
Name: airline_sentiment, dtype: float64
```

*Normalized Sentimate in the Training Set*

With Negative being about 3 times higher than neutral and 4 times higher than positive, I chose these as my respective weights. I think passed the weights into the data frame using the apply method so that any positive sentiment would have a 4, neutral a 3, and negative a 1.

I then called the sample method. I requested 18,000 records enough that the majority of negative sentiment would be passed along. In addition to the record total record count, I specified the weights column be used when picking records, along with setting replacement

to true. Lastly in the argument I set random_state equal to true, so anyone reproducing this analysis will get the same recordset.

Here were my results by record:

```
positive    6081
neutral     5967
negative    5952
Name: airline_sentiment, dtype: int64
```

While not exactly 6,000 each all three a very close to that number.

The next step was to tokenize the text in each of the tweets. For this the natural language tool kit (NLTK) was used and regular expression looking for word boundaries was used to identify each token in the record. I checked the length of the respective list and still had 18,000.

Next was to add the sentiment back on to the tokenized lists. This was done with an append method in for loop so that each record would get applied the sentiment tag. I printed out the first tupple to inspect. First the first element of the tuple was the list of the tokenized text, and the second was the sentiment tag.

The next step in the code was to sort the documents. The original data set from twitter was sorted by sentiment. In theory since we sampled records at random this step is likely not needed, but as my experience with Python is limited to 1 other class, I left this step in to be safe. As if the document were still sorted, the classifier we're getting ready to train would likely learn the pattern of where the sentiment tags are relative to the overall data.

Next, we ran a for loop to pull all the words from documents. Next used NLTK's frquency distribution to see how frequent the words were seen across the corpus. We then filtered documents into a new variable called word features and captured the most frequent 2000 words,

2000 is often used as the starting feature set in NLP for training a classifier. More can be added but risk over training the model. In addition, I elected not change the number as the twitter use more slang terms and given the period my test corpora were written, these additional terms likely would not have an impact on the model.

That said we still want to make sure we're finding words in the documents themselves. So, for the next step created a function called document features, which checks if the words captured in the previous step appeared in the respective document. Printed out the first document and glanced over and saw a few words equal true, so we know they're in the document and that the function preforms as expected.

Finally, after all the prep work, we're in a place where we can build the model. I imported the sklearn package's k-fold function so we can train with cross validation.

Cross Validation is a measure that splits the training dataset into equal pieces, usually 10, and then uses 90% of the data to train the model and 10% to test how the model is preforms. It then repeats this will a different split until all pieces have been tested.

For the first model build, we used a Naive Bayes Classifier with single word features. I elected to adjust the code to 90/10 cross validation. The unigram classifier as it is know preformed with 76.8% accuracy after testing all pieces, not a bad model.

Next, we tested bigrams. In order to do this, we created new function called alpha that searched for words in all words that started with an alpha character. This would remove numbers or unusual characters that were seen in the tweets. Next, we called the bigram associative measures from nltk. This will allow us to apply alpha as part of the bigram builder and get back bigrams that meaningful. In addition to alpha, we also applied a stopword remover. This will stop bigrams from being created with stop words included. So "the dog" will not be a bigram we're scanning. In addition to using, it through bigram measure nltk keeps track where the stop words were so that bigrams don't create by the two words on either of the stop word. In example "I walked the dog", by removing the in this wasy "walked dog" will not become a bigram in list.

Lastly, we captured the frequency of the bigrams and similar to the steps with the unigrams kept the first 2000 of them. Checked to see that document in th test set contained the bigrams, this time a bit more sparce.

At this point reran Naive Bayes with the bigrams and 10-fold cross validation. The model preforms significantly worse at 60.8%, so for the purpose of tagging out Corpora using the models build we will go with the unigram model.

## Applying the model

Now that the model has been built, the next step is to use it to classify each of the sentences in the test corpora.

In python we started by declaring empty lists to collect positive, negative, and neutral sentences that the classifier identifies. Although not used anywhere else in the script there was a call to collect the number of sentences classified into each group. This can quickly and easily be checked by calling the len argument. A sentences variable is created and passed to it is the test corpus (I ran for the first book and then had the code rerun for a the second). Lastly a positive count, negative count, and neutral count variables are created and set to zero.

I did make two modifications to this code. Sentences originally took the who text and tokenized it into sentences, a step I had already done to the corpus at the time of calling it from the Gutenberg Library. Again, further down we start a for loop that passes the sentences and 2000-word features identified together into the document features functions looking to see if those words are found in the sentence. There was a step in here to word Tokenize but that has already been done and was not longer needed.

I had actually left it in during my first run, not only did extend the execution time of the code because each word was being tested like sentence, it also gave no results for grammar phrases in part three. Once I had removed it, it was good.

The classifier logic works as follows. It looks at the words in each sentence against the predefined unigram features and and determines either positive, negative, or neutral. The resulting sentence is passed to an if statement that checks the result and appends the sentence to the empty list defined based on the tag. It also increments the counts by one. In the end it appends the total positive, total negative, and total neutral counts.

Now that the sentences have been tagged, we can look at the sentiment in each of the corpora.

For Anna Karenina there are 5,450 sentences that were tagged as positive sentiment and 6,886 sentences that were tagged as negative sentiment. 4,449 were neutral. So, over all the Corpus in this model shows negative sentiment of the time than positive. I did find this a bit surprising given that it was supposed to be about changing times, but characters in the story are still highly influenced by the social class structure in Russia and it's hold, while loosening, on their daily lives.

For The Great Gatsby a similar break, down of sentences in the Corpora. Positive sentences are 812, Negative Sentences are 935 and neutral sentences are 690. The Sentiment still leans slight negative, however with 37% of the sentiment being negative vs 33% positive it is closer breakdown than Anna which shows 42% Negative to 32% Positive.

## Parts of Speech

To finish off the visit in the world of Sentimate, I looked at the common parts of speech that were getting tagged positive and negative. To do this, in the code, I passed the Positive and Negative sentences from the model through NLTK's part of speech tagger. This will take each of the words in the sentence and tag them with their repecitve part of speech.

Next REGEX was used to define the various pharases that we would look at starting with Adjective Pharses, then Noun Pharese, and Finally Verb Phrases.

I would define the part of speech in a variable and then pass the expression along with the sentences to the NLKT's chunk parser. The next step was to define a function that would then extract the words and leave the tags behind. Lastly taking the phrases to see which ones were most frequent in the text, using the NLTK's frequency distribution.

For Anna K here are the top 50 positive adjective phrases: [('very glad', 29), ('so much', 20), ('too much', 13), ('very nice', 10), ('very good', 7), ('very much', 7), ('so glad', 6), ('very grateful', 5), ('" Good', 4), ('so happy', 4), ('very bad', 4), ('Very good', 4), ('Very glad', 4), ('very instant', 4), ('even more', 4), ('very interesting', 4), ('hardly perceptible', 3), ('s good', 3), ('very intelligent', 3), ('so good', 3), ('once more', 3), ('very young', 3), ('very sweet', 3), ('s jealous', 3), ('most various', 3), ('ever so much', 3), ('more excited', 3), ('so sweet', 3), ('so little', 3), ('very best', 3), ('too gloomy', 3), ('very attractive', 3), ('so strange', 3), ('sly smile',

2), (‘scarcely perceptible’, 2), (‘too great’, 2), (‘m glad’, 2), (‘not true’, 2), (‘so stupid’, 2), (‘so nice’, 2), (‘not good’, 2), (‘very different’, 2), (‘very sorry’, 2), (‘too late’, 2), (‘not angry’, 2), (‘most favorable’, 2), (‘always busy’, 2), (‘so simple’, 2), (‘even greater’, 2), (‘as much’, 2)]

Top 50 Positive Adjective Phrases for Gatsby: [(‘so much’, 3), (‘very good’, 3), (‘“ Good’, 2), (‘too much’, 2), (‘so hot’, 2), (‘more riotous’, 1), (‘most domesticated’, 1), (‘rather hard’, 1), (‘as much’, 1), (‘as cool’, 1), (‘rather impressive’, 1), (‘” insisted’, 1), (‘forward unashamed’, 1), (‘very bad’, 1), (‘pretty cynical’, 1), (‘most advanced’, 1), (‘yard high’, 1), (‘faintly handsome’, 1), (‘rather wide’, 1), (‘very beautiful’, 1), (‘so remarkable’, 1), (‘surprisingly formal’, 1), (‘too big’, 1), (‘s so much’, 1), (‘absolutely real’, 1), (‘somewhere last’, 1), (‘“ Much’, 1), (‘understandingly—much more’, 1), (‘so intimate’, 1), (‘highly indignant’, 1), (‘most amazing’, 1), (‘right there … Good’, 1), (‘very little’, 1), (‘incurably dishonest’, 1), (‘just as careless’, 1), (‘very hard’, 1), (‘oddly familiar’, 1), (‘very interesting’, 1), (‘very careful’, 1), (‘very polite’, 1), (‘very sentimental’, 1), (‘slightly older’, 1), (‘absolutely perfect’, 1), (‘too late.’, 1), (‘Once more’, 1), (‘once more’, 1), (‘most grotesque’, 1), (‘too hospitable’, 1), (‘indirectly due’, 1), (‘so little’, 1)]

Gatsby being a much shorter novel has a significantly lower frequency numbers than Anna Karenina. I also think the sentiment classifier would need some more tuning if this project were to continue. While most of these Phrase look like they are things that would register as positive, I do see some items in both list that would register a negative to me such as incurably dishonest.

As for negative adjective sentiment. Here are the top 50 phrases from Anna Karenina:

[(‘so much’, 42), (‘so many’, 22), (‘very glad’, 14), (‘most important’, 13), (‘once more’, 12), (‘not so much’, 10), (‘very good’, 10), (‘so awful’, 10), (‘very nice’, 9), (‘too much’, 9), (‘very sorry’, 8), (‘so little’, 8), (‘as much’, 8), (‘very interesting’, 7), (‘utterly unable’, 7), (‘too late’, 6), (‘so good’, 6), (‘so happy’, 6), (‘very busy’, 5), (‘very much’, 5), (‘however much’, 5), (‘quite different’, 5), (‘very instant’, 5), (‘not afraid’, 5), (‘so glad’, 5), (‘still more’, 5), (‘more important’, 5), (‘so terrible’, 4), (‘very simple’, 4), (‘so difficult’, 4), (‘very well aware’, 4), (‘not nice’, 4), (‘very same’, 4), (‘not angry’, 4), (‘not last’, 4), (‘most difficult’, 4), (‘most intimate’, 4), (‘not wrong’, 4), (‘very true’, 4), (‘not interested’, 4), (‘well aware’, 4), (‘so simple’, 4), (‘painfully conscious’, 3), (‘most awful’, 3), (‘so fond’, 3), (‘still worse’, 3), (‘just as much’, 3), (‘much interested’, 3), (‘so important’, 3), (‘very best’, 3)]

And top 50 Negative Adjective Phrases from The Great Gatsby:

[(’’ t ‘, 126), (’” “ ‘, 37), (’old sport’, 11), (‘it. ”’, 7), (‘long time’, 7), (‘front door’, 7), (‘old sport.’, 6), (‘o’ ‘, 6), (’few minutes’, 5), (‘all. ”’, 5), (‘young men’, 4), (‘“ Look’, 4), (‘young women’, 3), (‘other girl’, 3), (‘now. ”’, 3), (‘several times’, 3), (‘marble steps’, 3), (‘next door’, 3), (‘“ Let’, 3), (‘bad driver’, 3), (‘other people’, 3), (‘first time’, 3), (‘ve heard’, 3), (‘“ Daisy’, 3), (‘small town’, 3), (‘yellow car’, 3), (‘young man’, 2), (‘few days’, 2), (‘little sinister’, 2), (‘neighbour’ ‘, 2), (’warm windy’, 2), (‘s— ”’, 2), (‘longest day’, 2), (‘hundred people’, 2), (‘other hand’, 2), (‘down. ”’, 2), (‘several people’, 2), (‘first names’, 2), (‘back door’, 2), (‘me. ”’, 2), (‘next remark’, 2), (‘good time’, 2), (‘“ Don’, 2), (‘several weeks’, 2), (‘many years’, 2), (’” “ Yes’, 2), (‘fifty years’, 2), (‘s Series’, 2), (‘“ Come’, 2), (‘white car’, 2)]

For Anna many of the identified negative adjectives at the top of list could be either. Since the sentiment is based on the sentence. Things like “so much” “love” could be viewed as positive and “so much” “animosity” could be viewed as negative. In contrast the Great Gatsby we see some of the items found like the tokens " ’ should be disregarded as they’re not actual words.

Moving on to noun phrases. I reused the chunk parser we defined earlier and changed the code to look for Adjective followed by Noun to capture these phrases.

Here are the top 50 positive noun phrases from Anna Karenina:

[(‘“ Yes’, 56), (’’ t ‘, 49), (’” “ ‘, 38), (‘“ Ah ‘, 24), (’’ s ‘, 19), (’great deal’, 19), (‘same time’, 13), (‘…. ”’, 13), (‘“ Come’, 13), (‘“ Very’, 12), (‘“ Thank’, 11), (‘“ Don’, 9), (‘next day’, 9), (‘little girl’, 8), (‘young man’, 8), (‘old man’, 8), (‘good spirits’, 7), (‘first time’, 7), (‘” “ Yes’, 7), (‘s voice’, 6), (‘“ Let’, 6), (‘strange feeling’, 6), (‘other side’, 6), (‘_ ”’, 6), (‘other people’, 6), (‘long while’, 6), (‘same thing’, 6), (‘” Kitty’, 6), (‘“ Alexey’, 6), (‘sick man’, 6), (‘young men’, 5), (‘little hand’, 5), (‘s wife’, 5), (‘” Levin’, 5), (‘m glad’, 5), (‘handsome face’, 4), (‘perceptible smile’, 4), (‘ve come’, 4), (‘now. ”’, 4), (‘it. ”’, 4), (‘s face’, 4), (‘last year’, 4), (‘s hand’, 4), (‘“ No’, 4), (‘nice person’, 4), (‘first moment’, 4), (‘s heart’, 4), (‘great thing’, 4), (’’ clock ‘, 4), (‘“ Kitty ’, 4)]

And the top 50 positive noun phrases from The Great Gatsby:

[(’’ t ‘, 34), (’” “ ‘, 14), (‘“ Look ‘, 4), (‘“ Come ‘, 4), (’Good night’, 3), (‘“ Daisy’, 3), (‘old sport’, 3), (‘“ Mr’, 3), (‘” Tom’, 3), (‘s voice’, 3), (‘… Hot’, 3), (‘“ Yes’, 2), (‘true. ”’, 2), (‘“ Mrs’, 2), (‘“ Sure’, 2), (‘“ See’, 2), (‘yourself. ”’, 2), (‘green light’, 2), (‘well. ”’, 2), (‘small investigation’, 2), (‘re crazy’, 2), (‘“ Jimmy’, 2), (‘infinite hope’, 1), (‘various delays’, 1), (‘great bursts’, 1), (‘fast movies’, 1), (‘familiar conviction’, 1), (‘high intention’, 1), (‘many other books’, 1), (‘enormous eggs’, 1), (‘domesticated body’, 1), (‘salt water’, 1), (‘great wet barnyard’, 1), (‘s mansion’, 1), (‘own generation’, 1), (‘sturdy straw-haired man’, 1), (‘hard mouth’, 1), (‘supercilious manner’, 1), (‘arrogant eyes’, 1), (‘enormous leverage—a’, 1), (‘few minutes’, 1), (‘sunny porch’, 1), (‘inside. ”’, 1), (‘high hallway’, 1), (‘bright rosy-coloured space’, 1), (‘French windows’, 1), (‘fresh grass’, 1), (‘little way’, 1), (‘few moments’, 1), (‘whole town’, 1)]

The positive noun phrases are a bit more interesting. While both books show similar apostrophe phrases we have seen with adjectives. We see the many common pharses in Anna are are associated with characters like Alexey, Kitty, and Levin, while the Great Gatsby associates more with lifestyle like green lights, fast movies, french windows etc.

Continuing into negative noun phrases for Anna Karenina:

[(’’ t ‘, 408), (’” “ ‘, 229), (‘“ Yes ‘, 51), (’same time’, 45), (‘…. ”’, 43), (‘old man’, 40), (’’ s ‘, 40), (’long while’, 39), (‘first time’, 33), (‘great deal’, 30), (‘it. ”’, 30), (‘sick man’, 28), (‘other people’, 26), (‘other side’, 25), (‘” Levin’, 25), (‘several times’, 23), (‘old prince’, 23), (‘young man’, 21), (‘same thing’, 19), (‘“ Come’, 19), (‘next day’, 18), (‘few words’, 15), (‘me. ”’, 13), (‘s eyes’, 13), (‘only thing’, 13), (’’ clock ‘, 12), (’little girl’, 12), (‘same way’, 12), (‘” “ Yes’, 12), (‘him. ”’, 11), (‘young men’, 11), (‘s voice’, 11), (‘own room’, 10), (‘whole life’, 10), (‘m

afraid', 10), ('" Alexey', 10), ('previous day', 10), ('same position', 10), ('you. "', 9), ('o' ', 9), ('s face', 9), ('" Don', 9), ('s hand', 8), ('next room', 8), ('young people', 8), ('highest society', 8), ('now. "', 8), ('first minute', 7), ('many times', 7), ('" Kitty', 7)]

Nltk's Parts of Speech tagger seems to treat the letters on the right side of a contraction as nouns and apostrophe itself as an adjective giving at times some meaning less phrases like the second most common one found in Anna K. We do interestingly enough see both Kitty and Alexey associated with Negative Sentiment as well. If I were to continue doing more research on these novels I will probably look at if this shift happens in one of the sections of the book. As these characters were commonly associated publicly with their affairs.

Here are the top 50 negative noun phrases in the Great Gatsby:

[('' t ', 126), ('" " ', 37), ('old sport', 11), ('it. "', 7), ('long time', 7), ('front door', 7), ('old sport.', 6), ('o' ', 6), ('few minutes', 5), ('all. "', 5), ('young men', 4), ('" Look', 4), ('young women', 3), ('other girl', 3), ('now. "', 3), ('several times', 3), ('marble steps', 3), ('next door', 3), ('" Let', 3), ('bad driver', 3), ('other people', 3), ('first time', 3), ('ve heard', 3), ('" Daisy', 3), ('small town', 3), ('yellow car', 3), ('young man', 2), ('few days', 2), ('little sinister', 2), ('neighbour' ', 2), ('warm windy', 2), ('s— "', 2), ('longest day', 2), ('hundred people', 2), ('other hand', 2), ('down. "', 2), ('several people', 2), ('first names', 2), ('back door', 2), ('me. "', 2), ('next remark', 2), ('good time', 2), ('" Don', 2), ('several weeks', 2), ('many years', 2), ('" " Yes', 2), ('fifty years', 2), ('s Series', 2), ('" Come', 2), ('white car', 2)]

Gatsby's are a bit more pointed at things that a consumerist society would look down upon. Such as driving an old car, small towns, and the like. Interestingly that only one major Character in the book actually shows up on either list. Daisey who's affair leads to the death of both Gatsby and Tom at the end of the book.

Lastly, we have the verbs, which are the actions in the story. Here are Anna Karenina's top 50 verb phrases:

[('" answered', 18), ('ve been', 17), ('not be', 9), ('not help', 8), ('always did', 6), ('too was', 6), ('not understand', 5), ('not hear', 5), ('really was', 5), ('not go', 5), ('not know', 5), ('not believe', 4), ('" Delighted', 4), ('there was', 4), ('not speak', 4), ('not come', 4), ('always been', 4), ('ll be', 4), ('not have', 4), ('just been', 4), ('vividly recalled', 3), ('always excited', 3), ('not resist', 3), ('ve found', 3), ('not sleep', 3), ('not been', 3), ('never seen', 3), ('not care', 3), ('not feel', 3), ('always felt', 3), ('ever seen', 3), ('ll go', 3), ('not exactly ashamed', 3), ('not love', 2), ('once turned', 2), ('long known', 2), ('completely recovered', 2), ('not making', 2), ('well known', 2), ('not knowing', 2), ('not taking', 2), ('scarcely remembered', 2), ('once got', 2), ('involuntarily listening', 2), ('perfectly composed', 2), ('only just beginning', 2), ('particularly liked', 2), ('not expect', 2), ('not refuse', 2), ('not dancing', 2)]

Here are the top 50 positive Verbs for the Great Gatsby:

[('never loved', 6), ('ll be', 4), ('ve been', 4), ('just shows', 3), ('ever seen', 2), ('just got', 2), ('d seen', 2), ('casually conferred', 1), ('always leaning', 1), ('fragilely bound', 1), ('ever get', 1), ('rather surprised', 1), ('not mentioned', 1), ('ve gotten', 1), ('apparently finding', 1), ('s going', 1), ('" corroborated', 1), ('already crumbling', 1), ('just meant—', 1), ('around

looking', 1), ('just slip', 1), ('almost made', 1), ('actually invited', 1), ('never care', 1), ('always have', 1), ('somewhat drunk', 1), ('Absolutely real—have', 1), ('somewhere before.', 1), ('then concentrated', 1), ('attractively tight', 1), ('suddenly standing', 1), ('not confined', 1), ('slightly raised', 1), ('just heard', 1), ('suddenly there seemed', 1), ('always gathered', 1), ('continually breaking', 1), ('then came', 1), ('ever set', 1), ('nose flashed', 1), ('then added', 1), ('effectually prevented', 1), ('now assumed', 1), ('once ceased', 1), ('nearly toppled', 1), ('now vanished', 1), ('almost touching', 1), ('still falling', 1), ('there was', 1), ('now decently clothed', 1)]

What is interesting about the verbs in both books is that they are quite low in frequency. The other thing I find interesting is that state of being verbs are not showing up with much sentiment. In both books we see variations as the second most frequent phrases, but after that the verb to be in any forms barely shows again.

Here are the top 50 negative verbs for Anna Karenina:

[('not be', 93), ('not know', 62), ('not been', 49), ('not help', 45), ('not have', 42), ('not go', 37), ('ve been', 35), ('not understand', 27), ('not want', 24), ('not come', 22), ('not see', 21), ('there was', 20), ('not knowing', 19), ('not believe', 19), ('not going', 19), ('not take', 19), ('always did', 17), ('not think', 17), ('" answered', 16), ('not speak', 16), ('not say', 15), ('not looking', 15), ('never seen', 15), ('not answer', 14), ('not get', 14), ('not care', 14), ('always been', 14), ('not seen', 14), ('long been', 13), ('not hear', 13), ('not give', 12), ('not let', 12), ('never have', 12), ('not tell', 12), ('not having', 12), ('m going', 12), ('never be', 11), ('never had', 11), ('only be', 11), ('not find', 11), ('not make', 10), ('not love', 9), ('not live', 9), ('not look', 9), ('never been', 9), ('not agree', 9), ('ll be', 8), ('always had', 8), ('t be', 8), ('not admit', 8)]

Here are the top 50 verbs for the Great Gatsby:

[('ve been', 9), ('m going', 8), ('ll be', 5), ('there was', 5), ('d been', 5), ('d be', 2), ('always watch', 2), ('then miss', 2), ('about was', 2), ('ll do', 2), ('m having', 2), ('re going', 2), ('ever knew', 2), ('never seen', 2), ('d never seen', 2), ('not been', 2), ('t be', 2), ('always been', 2), ('hardly knew', 2), ('never told', 2), ('then turned', 2), ('never loved', 2), ('s going', 2), ('never seemed', 2), ('m inclined', 1), ('also made', 1), ('unjustly accused', 1), ('snobbishly suggested', 1), ('never found', 1), ('ever find', 1), ('temporarily closed', 1), ('re descended', 1), ('never saw', 1), ('finally said', 1), ('just left', 1), ('more recently arrived', 1), ('much more successfully looked', 1), ('really begins', 1), ('ever played', 1), ('afterward savours', 1), ('enormously wealthy—even', 1), ('rather took', 1), ('then drifted', 1), ('scarcely knew', 1), ('always had', 1), ('then rippled', 1), ('just been', 1), ('almost surprised', 1), ('so much wanted', 1), ('ve heard', 1)]

The contrast between positive and negative for both books when it comes to verbs is a bit interesting. For one verb phrases associated with a negative sentence are more frequent, suggesting when the writers choose to be negative, they employ less use of vocabulary. In addition, State of being verbs rise to the top of the negative verb phrases.

## Creation of Data Frame

Now that we're done with looking at the outcomes for Anna Karenina and The Great Gatsby with all the sentences tagged with sentiment, the last step is to create a Data Frame for the professor's review. The file could also be used to review the algorithm's tagging and edited for the use in future tagging. Though for the course this is likely the end of the Journey.

In order to create the file, at the end of the Anna Karenina Section, I created an empty list, following the example in the instructor provided code. Since my text was one large corpus, I wrote three loops that took the sentences that had previously been tagged into positive, negative, and neutral. Each sentence was passed into its own dictionary, with the sentiment, name of the book, author, and total number of positive, negative, and neutral sentences in the book. Each resulting dictionary was then appended to the empty list.

I repeated the same thing for The Great Gatsby appending each sentence in dictionary to the same list that I started in the Anna Karenina section of my code. I then used pandas DataFrame function and convert the list of dictionaries to a DataFrame. At this point I checked the first five records which would all be positive sentences from Anna Based on how the loop was written and the last 5 sentences all neutral sentences from Gatsby.

| | Text | Sentiment | Book | Author | Total Postive Sentences | Total Negative Sentences | Total Neutral Sentences |
|---|---|---|---|---|---|---|---|
| 0 | [[, Illustration, ], ANNA, KARENINA, by, Leo, ... | Positive | Anna Karenina | Leo Tolstoy | 5635 | 6836 | 4433 |
| 1 | [Everything, was, in, confusion, in, the, Oblo... | Positive | Anna Karenina | Leo Tolstoy | 5635 | 6836 | 4433 |
| 2 | [", Now, ,, how, was, it, ?] | Positive | Anna Karenina | Leo Tolstoy | 5635 | 6836 | 4433 |
| 3 | [Yes, ,, but, then, ,, Darmstadt, was, in, Ame... | Positive | Anna Karenina | Leo Tolstoy | 5635 | 6836 | 4433 |
| 4 | [Stepan, Arkadyevitch, ', s, eyes, twinkled, g... | Positive | Anna Karenina | Leo Tolstoy | 5635 | 6836 | 4433 |

*First 5 records Postivie Anna Sentences*

| | Text | Sentiment | Book | Author | Total Postive Sentences | Total Negative Sentences | Total Neutral Sentences |
|---|---|---|---|---|---|---|---|
| 19336 | [By, God, it, was, awful—, ", I, couldn, ', t,... | Neutral | The Great Gatsby | F Scott FitzGerald | 817 | 948 | 672 |
| 19337 | [One, night, I, did, hear, a, material, car, t... | Neutral | The Great Gatsby | F Scott FitzGerald | 817 | 948 | 672 |
| 19338 | [Then, I, wandered, down, to, the, beach, and,... | Neutral | The Great Gatsby | F Scott FitzGerald | 817 | 948 | 672 |
| 19339 | [And, as, the, moon, rose, higher, the, inesse... | Neutral | The Great Gatsby | F Scott FitzGerald | 817 | 948 | 672 |
| 19340 | [And, as, I, sat, there, brooding, on, the, ol... | Neutral | The Great Gatsby | F Scott FitzGerald | 817 | 948 | 672 |

*Last 5 Records Neutral Gatsby Sentences*

I the used the df.to_csv to create a CSV file of the data and opened the CSV in excel to ensure those sentences matched between what Python was showing and what was output on the CSV.

These matched including the record numbering from Python. Quotation marks seem to have converted poorly, but overall, I would its success.

| | Text | Sentimen | Book | Author | Total Post | Total Neg | Total Neut |
|---|---|---|---|---|---|---|---|
| 0 | ['[', 'Illustration', ']', 'ANNA', 'KARENINA', 'by', 'Leo', 'Tolstoy', 'Translated', 'by', 'Constance', 'Garnett', 'Con | Positive | Anna Kare | Leo Tolstc | 5635 | 6836 | 4433 |
| 1 | ['Everything', 'was', 'in', 'confusion', 'in', 'the', 'Oblonskys', 'â€™', 'house', '.'] | Positive | Anna Kare | Leo Tolstc | 5635 | 6836 | 4433 |
| 2 | ['â€œ', 'Now', ',', 'how', 'was', 'it', '?'] | Positive | Anna Kare | Leo Tolstc | 5635 | 6836 | 4433 |
| 3 | ['Yes', ',', 'but', 'then', ',', 'Darmstadt', 'was', 'in', 'America', '.'] | Positive | Anna Kare | Leo Tolstc | 5635 | 6836 | 4433 |
| 4 | ['Stepan', 'Arkadyevitch', 'â€™', 's', 'eyes', 'twinkled', 'gaily', ',', 'and', 'he', 'pondered', 'with', 'a', 'smile', '.'] | Positive | Anna Kare | Leo Tolstc | 5635 | 6836 | 4433 |

*CSV first five records*

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 19336 | ['By', 'God', 'it', 'was', 'awfulâ€™', 'â€', 'I', 'couldn', 'â€™', 't', 'forgive', 'him', 'or', 'like', 'him', ',', 'but', 'I', 'saw' | Neutral | The Great | F Scott Fit | 817 | 948 | 672 |
| 19337 | ['One', 'night', 'I', 'did', 'hear', 'a', 'material', 'car', 'there', ',', 'and', 'saw', 'its', 'lights', 'stop', 'at', 'his', 'front', | Neutral | The Great | F Scott Fit | 817 | 948 | 672 |
| 19338 | ['Then', 'I', 'wandered', 'down', 'to', 'the', 'beach', 'and', 'sprawled', 'out', 'on', 'the', 'sand', '.'] | Neutral | The Great | F Scott Fit | 817 | 948 | 672 |
| 19339 | ['And', 'as', 'the', 'moon', 'rose', 'higher', 'the', 'inessential', 'houses', 'began', 'to', 'melt', 'away', 'until', 'gra | Neutral | The Great | F Scott Fit | 817 | 948 | 672 |
| 19340 | ['And', 'as', 'I', 'sat', 'there', 'brooding', 'on', 'the', 'old', ',', 'unknown', 'world', ',', 'I', 'thought', 'of', 'Gatsby', | Neutral | The Great | F Scott Fit | 817 | 948 | 672 |

*CSV last five records*

I also ran a pivot table to check that the number of records were matching to totals calculated in Python and reported on the data set. All looks good. But note the way the output was designed each record in the CSV tells total number of Positive, Negative, and Neutral Sentences in the book, I had to change the aggregation to max other wise it would be adding it over all the records and reporting numbers way of line. In a sense the left part of the CSV records is a granular look, while the right is the meta data about the overal text itself.

| Count of Text | | Sentiment | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Book | Author | Negative | Positive | Neutral | Book | Author | Max of Total Postive Sentences | Max of Total Negative Sentences | Max of Total Neutral Sentences |
| Anna Karenina | Leo Tolstoy | 6836 | 5635 | 4433 | Anna Karenina | Leo Tolstoy | 5635 | 6836 | 4433 |
| The Great Gatsby | F Scott FitzGerald | 948 | 817 | 672 | The Great Gatsby | F Scott FitzGerald | 817 | 948 | 672 |
| Grand Total | | 7784 | 6452 | 5105 | Grand Total | | 5635 | 6836 | 4433 |

## Conclusion

After balancing the training set and applying it to both books, we still see diversion towards negative sentiment being the main driver. This was a bit more prevalent in Anna than it was in Gatsby. Interestingly with the verbs just “being” is somehow seen as negative. If I were to continue with these texts I would probably start by trying to manually review the outcomes and seeing if I agree with the sentiment that that machine learning model predicted, but as we saw with the Airline sentiment before balancing people tend to be more driven to respond to negative events than positive ones, so this is likely true even in novels and surprisingly has changed more 75 years since these books were written.